Trends in Intelligent System Technologies for Airfoil Design: - Computer Algebra Techniques for Gradient Computation through the Continuous Adjoint Approach - Use of Approximate Fitness Evaluators in Evolutionary Optimization

Domenico Quagliarella[‡], **Domenic D'Ambrosio**[⋆], **Angelo Iollo**[⋆]
[‡] Centro Italiano Ricerche Aerospaziali
Via Maiorise, 81043 Capua, Italy
[⋆]Dipartimento di Ingegneria Aeronautica e Spaziale, Politecnico di Torino
C.so Duca degli Abruzzi 24, 10129 Torino, Italy
d.quagliarella@cira.it, domenic@athena.polito.it, iollo@polito.it

1 Introduction

Aerodynamic shape design is a challenging application for modern computational fluid dynamic. This is, indeed, a very productive field, both in term of scientific publications and developed applications. Furthermore it is possible to find excellent analysis/design software released in the public domain (See for example [1, 2]). However, industry, in particular automotive and aerospace, but also emerging fields like wind and sea extracted energy, continuously needs increased product performance and competitiveness. This requires increasingly refined, advanced and complex computational models capable of describing flow behavior with higher and higher fidelity.

Thus the design tools available to scientist and engineers are challenged by the growing complexity of simulation models. The increasing price to performance ratio of nowadays computers is a strong allied of designers in this though race, but often it is not enough.

Objectives of this lecture is to explore and analyze some of the technologies that, together with the increasingly available computational power, may help to exploit the tremendous potential of high fidelity flow field models to aerodynamic shape design challenges.

For the sake of simplicity we will concentrate on aerofoil design. Two aspects will be considered: the computer algebra assisted development of code related to gradient computation through adjoint techniques and the use of fitness approximation techniques to improve the performance of search procedures. The concern of this first lecture will be more methodological, while some applications will be illustrated in the next one.

The outline of the first first part of this lecture is as follows: some short remarks on adjoint problems will introduce the framework in which the Computer Algebra techniques will be mainly used here. Then a brief outline of symbolic computation systems will be given, with particular reference to the features that are here used for adjoint equation derivation. The implemented algorithm will be described and illustrated with an example on a simple problem. The adjoint and the related boundary conditions for 3D Navier-Stokes are then reported, while an example of derivation is given in appendix A.

The second part of this lecture will focus on the use of approximated fitness evaluators. The problem will be analyzed using the framework of multi-objective optimization, and some possible approaches to the problem of mixing exact and approximated evaluations will be discussed.

2 Adjoint formulation for gradient evaluation

The adjoint formulation appears in optimal control theory as a tool to compute the constrained gradient to a given functional. In fluid mechanics the applications span from shape design to flow control [3, 4, 5, 6]. The

Paper presented at the RTO AVT Course on "Intelligent Systems for Aeronautics", held in Rhode-Saint-Genèse, Belgium, 13-17 May 2002, and published in RTO-EN-022.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 JUN 2003	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Trends in Intelligent SystemTechnologies for Airfoil Design: - Computer Algebra Techniques for Gradient Computation through the Continuous Adjoint Approach - Use of Approximate Fitness Evaluators in Evolutionary Optimization	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Centro Italiano Ricerche Aerospaziali Via Maiorise, 81043 Capua, Italy**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM001519. RTO-EN-022, The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**26**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

adjoint method is based on the solution of an additional set of equations for the Lagrange multipliers. The Lagrange multipliers local value is a measure of the functional sensitivity to the flow variables local variation. This provides, for example, a quantitative criteria for grid refinement [7].

In mathematical terms the adjoint approach to gradient computation can be concisely derived from the Lagrange identity:

$$(Av, w) = (v, A^* w) \tag{1}$$

where $(\quad , \quad)$ is the scalar product in the appropriate Hilbert space. This identity defines the adjoint operator A^* when a linear or non-linear operator A and when $v \in D,\ w \in D^*$ are given. Now, if

$$J = (p, v) \tag{2}$$

is a given functional and

$$Av = f, \quad A^* w = p \tag{3}$$

are, respectively, the state and the adjoint equations, then it is possible to write

$$J = (p, v) = (f, w) \tag{4}$$

as a direct consequence of identity 1. This means that J can be expressed either in terms of the state vector v or in terms of the adjoint one w. The small perturbations theory can then be used to obtain the functional variation for non-linear state equations. In general, if $A(v)$ is an operator that depends non-linearly on v, it is possible to write, under the hypothesis of small perturbations, that

$$\begin{aligned} f &= f_0 + \varepsilon f_1 + \dots \\ v &= v_0 + \varepsilon v_1 + \dots \\ A(v_0 + \varepsilon v_1 + \dots) &= A(v_0) + \varepsilon \left.\frac{dA}{dv}\right|_0 v_1 + \dots \end{aligned} \tag{5}$$

Therefore the non-linear state equation $A(v)v = f$ can be decomposed as follows

$$\begin{aligned} &A(v_0)v_0 = f_0 \\ &A(v_0)v_1 + \left(\left.\frac{dA}{dv}\right|_0 v_1 \right) v_0 = f_1 \end{aligned} \tag{6}$$

From the last equation descends immediately the first order linearized operator:

$$A_1(v_0)\bullet = A(v_0)\bullet + \left(\left.\frac{dA}{dv}\right|_0 \bullet \right) v_0 \tag{7}$$

In the same way, it is possible to write the functional J as

$$J = J_0 + \varepsilon J_1 \tag{8}$$

where

$$\begin{aligned} &J_0 = (v_0, p) = (w_0, f_0) \\ &\delta J = J_1 = (v_1, p) = (w_1, f_1) \\ &\text{with} \quad A_1^*(v_0) w_1 = p \end{aligned} \tag{9}$$

In other words we obtain explicitly the functional variation with respect to the right hand side (RHS) of the governing equation, i.e., the gradient of J with respect to f. Hence, from a mathematical point of view, the derivation of the adjoint state equations for a given functional J is a straightforward process. A disadvantage of this method is that complicated functionals often require a lengthy and error-prone manual derivation and hand coding process. In this respect Computer Algebra (CA) techniques can be profitably used to ease and speed-up this process. In the following sections such technique will be described and then used to automatically obtain the adjoint equations and pertinent boundary conditions of the Navier-Stokes equations. Gradient expression is also automatically computed for a generic objective function. A standard computer algebra language (MAPLE [8]) is used to build the automated derivation procedure.

3 Automatic derivation of the adjoint system and gradient

3.1 General remarks on symbolic computation

Computer algebra systems for symbolic computation have a number of peculiarities that make them radically different from numeric computing tools. Indeed, they operate on symbols (the rich set of mathematical objects), rather than numbers as classic numerical programming languages. The mathematical objects are transformed following exact equivalence rules, and there are not precision limits in the representation of numbers. The algebraic manipulation algorithms can be very complex and sophisticated so that a computer algebra language can be considered a way of storing and efficiently using the knowledge and expertise of thousands of brilliant mathematicians. This allows an easy and straightforward use of computer algebra systems as "scratchpad" tools in the interactive solutions of complex mathematical problems.

On the other hand, there are some drawbacks that have to be taken into account when developing complex applications. In particular, there are cases in which the output is very complex, and considerable effort has to be devoted to keeping it easily understandable. Furthermore, it has to be also considered that it is not easy to learn how to write efficient code in a computer algebra language, because the programming model is fundamentally different with respect to "numerical programming", with recursion, list and set operations, and expression identification and and matching on a complex and rich set of objects.

The approach followed in the present work was the research of a compromise between output conciseness and ease of implementation, and the main steps of the developed adjoint equation derivation procedure can be summarized as follows:

- The Navier-Stokes equations have been specified term by term, but the volume and surface integrals of the Lagrangian have been left unexpanded.
- The classical manipulations for adjoint derivation were obtained using Green formulas, while the variation with respect to geometry were obtained through Hadamard formulas.

This approach required the capability of discriminating between the various possible structures of a given Lagrangian, in order to apply the proper transformations. This was obtained applying recursively the pattern matching functions of the CA system.

3.2 Adjoint computation

We provide here expressions and identities that will be useful for the automatic computation of the integral variations by the MAPLE V computer algebra system [8].

The first step in the automated derivation process is the expression of the flow field equation in indexed form, using, in the present implementation, the Grtensor software [9] for tensor calculus. The expanded equation and boundary conditions, referenced as $\mathbf{R}$, are then combined with the objective function $\mathbf{O}$ to form the Lagrangian equation

$$\begin{aligned} L \;=\; & \iint_{\Sigma(h)} \mathbf{O}(x,\ldots,u,u_x,u_y,u_z,u_{xx},u_{xy},\ldots,h,\ldots)\; d\sigma + \\ & \iiint_{\Omega(h)} \lambda\mathbf{R}(x,\ldots,u,u_x,u_y,u_z,u_{xx},u_{xy},\ldots,h,\ldots)\; d\Omega \end{aligned} \tag{10}$$

The adjoint equations and boundary conditions are obtained computing the variation of L with respect to the flow field unknowns u, while the gradient has been computed through the variation of L with respect to the boundary defining functions h. The formulas used for the variation of volume and surface integrals are reported in the following sections. One of the tricky aspects of this approach is that the variation of L has to be computed with respect to quantities that are, in turn, functions of the coordinates x, y and z. In fact, standard MAPLE differentiation function is unable to perform variational calculus. Yet, there are ways to get around such a difficulty,

for example using the `pdiff` [10] function, available in MAPLE V share library. Such function enables the computation of expressions such as

$$\frac{\partial f(x, u(x))}{\partial u(x)}$$

that are not computable with the standard MAPLE differentiation function. The Green's theorem is then repeatedly applied to move the terms containing the derivative of a variation from the volume terms to the surface integral terms. MAPLE pattern matching functions have been extensively used both for the variation procedure and the Green's theorem procedure implementation.

3.3 Symbolic computation algorithm

Given the Lagrangian expression, in the following we give some details about the algorithm used to obtain the adjoint equations, the boundary conditions, and the gradient expression. The derivation of the formulas adopted to express the variation of the functionals is reported in appendix B.

The first MAPLE function implemented is called `variation`. This function takes as input a generic functional L, a function u to be variated and eventually the vector function $\mathbf{h}$ that defines the geometric deformation. In general, the expansion obtainable for a generic variation is of the kind

$$\delta L_u(x, \ldots, u, u_x, u_y, u_z, u_{xx}, u_{xy}, \ldots, h, \ldots) =$$

$$\frac{\partial L}{\partial u}\delta u + \frac{\partial L}{\partial u_x}\delta u_x + \frac{\partial L}{\partial u_y}\delta u_y + \frac{\partial L}{\partial u_z}\delta u_z + \frac{\partial L}{\partial u_{xx}}\delta u_{xx} + \frac{\partial L}{\partial u_{xy}}\delta u_{xy} \tag{11}$$

When the variation is computed with respect to the geometry, a pattern matching algorithm checks each term of the functional L in order to find volume or surface integrals. In the case of a surface integral, then the following expression is used to compute the variation:

$$\delta F = \iint_S \nabla f \cdot \frac{\partial \mathbf{h}}{\partial g}\, \delta g \, d\sigma + \iint_S f \boldsymbol{\mathcal{H}} \cdot \frac{\partial \mathbf{h}}{\partial g}\, \delta g \, d\sigma + \iint_S \frac{\partial f}{\partial g}\, \delta g \, d\sigma \tag{12}$$

Appendix sub-section B.1 reports the derivation of this expression and the meaning of the terms involved. In the case of volume integrals, the formula

$$\delta F = \iiint_T \frac{\partial f}{\partial g}\, \delta g \, dV + \iint_{FT} f \, \frac{\partial \mathbf{h}}{\partial g} \cdot \mathbf{n}\, \delta g \, d\sigma \tag{13}$$

is adopted. Expression derivation and terms definition are in B.3. These formulas are exactly equivalent to the Hadamard formulas. It is worth to remark here that volume and surface integrals, need special treatment only when the integration manifold is given in implicit form. In the other cases they can be reduced to equivalent double and triple definite integrals and treated with formula 11.

The second MAPLE function implemented is called `intparts3d`. The input to this function is the variated functional δL, and in output an equivalent form of the functional δL is generated, were the terms containing derivatives of the variation are moved to surface integrals. It works applying recursively the integration by parts and the Green's theorem to the volume integral terms containing derivatives of the variation, e.g. δu_x or δu_{xy}. The resulting functional variation presents terms containing derivatives of the variation only inside surface integrals. In a subsequent step, these terms can be eliminated through the imposition of suitable boundary conditions. This last step, currently, is not automated and require some care and attention on the part of the user. To help this process, the functions `extract_adj_eq` and `extract_adj_bceq` have been introduced. They extract the adjoint equations and boundary conditions, respectively, again by pattern recognition. Finally, two interface procedures, `compute_adj_eq` and `compute_adj_eq_bc`, heve been introduced to manage the whole process of adjoint equation and boundary conditions derivation. The listing in MAPLE code of the first one is reported below.

```
compute_adj_eq :=
   proc(lagrangian, v, dv, n_intp, h, verbose_level, aliasing::boolean)
   option remember;
   local lg_v, i, c1, sp, adj_eq;
       lg_v := variation(lagrangian, [v=v+dv], h, [s1, s2], [x1, x2, x3]);
       lg_v := expand(lg_v);
       if aliasing then autoAlias(lg_v) fi;
       if verbose_level >= 2 then print("LAGRANGIAN"); print(lg_v) fi;
       c1 := lg_v;
       for i from 1 by 1 to n_intp do
           c1 := intparts_3d(c1,dv);
           c1 := expand(convert(c1,diff));
           if aliasing then autoAlias(c1) fi;
           if verbose_level >= 1 then
               print("INTEGRATION BY PARTS N.", i); print(c1)
           fi;
       od;
       adj_eq := extract_adj_eq(c1,dv);
   end:
```

The equations obtained by the automated procedure are of course less concise compared to those obtained manually. Indeed, the governing equations are fully expanded as a first step in the variation computation, to ease the pattern matching process. Furthermore, the comparison, useful for debug purposes, with the manually obtained equations is complicated by some asymmetries that may be found in the automatically derived terms. This is due to the following identity (see sub-appendix B.4 for details):

$$\iiint_{\Omega} v_{xy}\, d\Omega = \iint_{\Sigma} v_x n_2\, d\Sigma = \iint_{\Sigma} v_y n_1\, d\Sigma \tag{14}$$

This comes from the application of the divergence theorem to terms containing crossed derivatives. Indeed, the theorem can be applied here in two different ways, and the automatic procedure does not know which form has to be chosen to keep symmetry. However, the same identity 14 can be applied to suitably transform the obtained expressions.

3.4 A simple application example to potential flows

An example of adjoint equation derivation is here given to show how the developed MAPLE code may be used. The reader may refer to appendix A for the application to Navier-Stokes equations.

A symmetric airfoil is immersed in a potential flow at zero angle of attack (see figure 1), and a symmetric transpiration velocity W is assigned in the direction normal to the airfoil surface. The governing equations are therefore:

$$\nabla^2\phi = 0 \quad \text{in the field,} \qquad \nabla\phi \cdot \mathbf{n} = W \quad \text{on the airfoil} \tag{15}$$

The problem is to find a transpiration velocity distribution W^* such that a given target potential Φ is matched. Hence the objective function that will be minimized is:

$$\frac{1}{2} \oint (\phi - \Phi)^2\, d\sigma \tag{16}$$

on the airfoil surface. The current version of adjoint derivation code was developed taking into account three-dimensional fields; therefore in the following example, the two-dimensional field will be treated as 3D field with zero gradient of the flow variables in $x3$ direction. In the same spirit the airfoil is a cylindric surface identified by two curvilinear coordinates $s1, s2$. Thus we will have surface integrals instead of contour integrals on the boundary and volume integrals in the field.

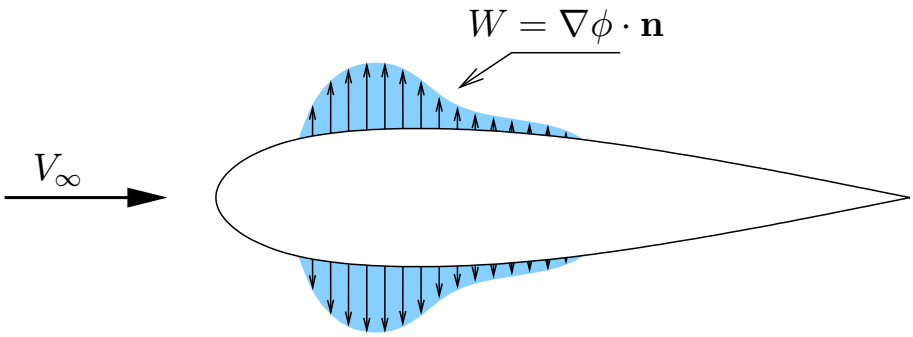

Figure 1: 2D symmetric potential flow with blowing.

Here the MAPLE interactive session for the adjoint computation follows. The `alias_dependencies` function set a functional dependence for a given list of variables; dW is the increment of the blowing velocity, y is the airfoil ordinate, $x1, x2, x3$ are the field coordinates, while $s1, s2$ are the surface coordinates. Note that to have the procedure working properly, we had to specify explicitly the dependence of ϕ and of its increment $d\phi$ on field and surface coordinates. The same dependence is specified for the vector normal to the airfoil surface $\mathbf{n} = [n1, n2, n3]$ and for the deformation field $\mathbf{h} = [h1, h2, h3]$ (not explicitly used here).

```
> alias_dependencies([y,W,dW],[s1]):
> varlist := [x1,x2,x3,s1,s2]:
> alias_dependencies([phi,dphi,n1,n2,n3,h1,h2,h3],varlist):
> alias_dependencies([Phi],[s1,s2]):
> Obj:=Int(Int(1/2*(phi-Phi)^2,s1),s2);
```

$$Obj := \iint \frac{1}{2}\,(\phi - \Phi)^2\, ds1\, ds2$$

```
> Potential_eq:=diff(phi,'$'(x1,2))+diff(phi,'$'(x2,2)):autoAlias(%);
```

$$\phi_{x1,\,x1} + \phi_{x2,\,x2}$$

```
> n:=[n1,n2];
```

$$n := [n1,\ n2]$$

```
> Neumann_bc:=dotprod(grad(phi,[x1,x2]),n,'orthogonal')-W:
> autoAlias(%);
```

$$\phi_{x1}\ n1 + \phi_{x2}\ n2 - \mathrm{W}$$

```
> alias_dependencies([eta,beta],varlist):
> Lagrangian:=Obj+Int(Int(Int(eta*Potential_eq,x1),x2),x3)+
> Int(Int(beta*Neumann_bc,s1),s2);
```

$$Lagrangian := \iint \frac{1}{2}\,(\phi - \Phi)^2\, ds1\, ds2 + \iiint \eta\,(\phi_{x1,\,x1} + \phi_{x2,\,x2})\, dx1\, dx2\, dx3$$
$$+ \iint \beta\,(\phi_{x1}\ n1 + \phi_{x2}\ n2 - \mathrm{W})\, ds1\, ds2$$

```
> adj_dphi:=compute_adj_eq(Lagrangian,phi,dphi,2,[h1,h2,h3],0,true);
```

$$adj\_dphi := \eta_{x1,\,x1} + \eta_{x2,\,x2}$$

```
> adj_dphi_bc:=compute_adj_bc(Lagrangian, phi, dphi, 2, [n1, n2, n3],
> [h1, h2, h3], 0, true);
```

$$adj\_dphi\_bc :=$$
$$\phi - \Phi + \frac{\eta\ dphi_{x1}\ n1}{dphi} - \eta_{x1}\ n1 + \frac{\eta\ dphi_{x2}\ n2}{dphi} - \eta_{x2}\ n2 + \frac{\beta\ n1\ dphi_{x1}}{dphi} + \frac{\beta\ n2\ dphi_{x2}}{dphi}$$

```
> adj_dphi_bc_airf:=subs(beta=-eta,adj_dphi_bc);
```

$$adj\_dphi\_bc\_airf := \phi - \Phi - \eta_{x1}\ n1 - \eta_{x2}\ n2$$

```
> c6:=variation(Lagrangian,[W=W+dW],[s1,y,s2],[s1,s2],[x1,x2,x3]);
```

$$-\iint \beta\ dW\ ds1\ ds2$$

In the above equation the gradient with respect to the blowing velocity W can be recognized as being given by the variation of the Lagrangian with respect to W.

4 The compressible Navier Stokes Adjoint Equations

In this section the mathematical adjoint to the Navier-Stokes equations are derived. The notation is slightly different from that of the previous chapter for the sake of conciseness and clarity. In particular, the variations will be indicated with a circumflex accent.

4.1 State equations

In this section we make the notation clear by giving the steady compressible Navier-Stokes equations under divergence form and indicial notation.

Conservation of mass

$$\frac{\partial \rho v_i}{\partial x_i} = 0 \tag{17}$$

Conservation of momentum

$$\frac{\partial}{\partial x_j}\left\{\rho v_i v_j + p\delta_{i\,j} - \mu\left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\,\delta_{i\,j}\right]\right\} = 0 \tag{18}$$

Conservation of energy

$$\frac{\partial}{\partial x_i}\left\{\rho v_i H - k\frac{\partial T}{\partial x_i} - \mu\left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\,\delta_{i\,j}\right] v_j\right\} = 0 \tag{19}$$

H is the total enthalpy per unit mass, k the heat conductivity, and the other symbols have their usual meaning.

Let Ω be the flow domain and Σ its surface. The no-slip boundary conditions are imposed on Σ, elsewhere we consider suitable far field conditions. The boundary condition on temperature is to be detailed later.

4.2 Lagrangian

I is a generic functional depending on the flow field and it is defined either over Ω or over Σ. We want to find a surface Σ such that the first order variation of I with respect to a perturbation of Σ is zero.

To this end, we introduce an auxiliary functional L defined as follows

$$\begin{aligned} L = I \quad &+ \quad \int_\Omega \eta\,\frac{\partial \rho v_i}{\partial x_i}\,d\Omega \\ &+ \quad \int_\Omega \lambda_i\,\frac{\partial}{\partial x_j}\left\{\rho v_i v_j + p\delta_{i\,j} - \mu\left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\,\delta_{i\,j}\right]\right\} d\Omega \\ &+ \quad \int_\Omega \zeta\frac{\partial}{\partial x_i}\left\{\rho v_i H - k\frac{\partial T}{\partial x_i} - \mu\left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\,\delta_{i\,j}\right] v_j\right\} d\Omega \\ &+ \quad \int_\Sigma \xi_i v_i\,d\Sigma \\ &+ \quad \int_\Sigma \tau\Theta(T)\,d\Sigma \end{aligned} \tag{20}$$

$\Theta(T) = T - T_w$ with prescribed boundary temperature, whereas for adiabatic flows $\Theta(T) = \partial T/\partial x_i n_i$. The auxiliary functions η, λ_i and ζ belong to $H^2(\Omega)$. $\xi \in L^2(\Sigma)$ and $\tau \in L^2(\Sigma)$ or $\tau \in H^1(\Sigma)$ according to the boundary conditions selected for temperature. We refer to all these functions as Lagrange multipliers.

The lagrangian function L transforms the constrained problem defined for I into an unconstrained problem: it is now required to find the flow field, Σ and the Lagrange multipliers in order that the respective first order variations of L are zero. The variation of L with respect to the Lagrange multipliers yields the flow equations and boundary conditions. We concentrate on the variation of L with respect to the conservative flow variables ρ, ρv_i, ρE, with E the total energy per unit mass.

Take the variation of the first integral in eq. 20, using the Gauss identity we have

$$\int_\Sigma \eta\, \widetilde{\rho v_i}\, n_i\, d\Sigma - \int_\Omega \widetilde{\rho v_i}\, \frac{\partial \eta}{\partial x_i}\, d\Omega \tag{21}$$

where n_i is the outward unit normal to Σ, and $\widetilde{\cdot}$ denotes the function variation.

Consider now the second integral in eq. 20 and apply the Gauss identity twice to get

$$\begin{aligned} &\int_\Sigma \lambda_i f_i\, d\Sigma - \int_\Omega (\rho v_i v_j + p\delta_{i\,j})\, \frac{\partial \lambda_i}{\partial x_j}\, d\Omega - \int_\Omega \frac{\partial}{\partial x_j} \left\{ \mu \left[\frac{\partial \lambda_i}{\partial x_j} + \frac{\partial \lambda_j}{\partial x_i} - \frac{2}{3} \frac{\partial \lambda_k}{\partial x_k}\, \delta_{i\,j} \right] \right\} v_i\, d\Omega \\ + \quad &\int_\Sigma \mu \left[\frac{\partial \lambda_i}{\partial x_j} + \frac{\partial \lambda_j}{\partial x_i} - \frac{2}{3} \frac{\partial \lambda_k}{\partial x_k}\, \delta_{i\,j} \right] v_i n_j\, d\Sigma \end{aligned} \tag{22}$$

where the components of the force per unit surface experienced at the boundary are

$$f_i = \left\{ \rho v_i v_j + p\delta_{i\,j} - \mu \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k}\, \delta_{i\,j} \right] \right\} n_j$$

To be remarked is that the linear second order adjoint operator keeps the same form of the viscous stresses.

In eq. 22 the flow variables do not appear under the form of conservative variables. However, in the first integral, for a reason that will be clear later, we leave the flow variation under the form $\widetilde{f_i}$, while in the others we write the variations in terms of conservative variables. For example we have

$$\widetilde{\rho v_i v_j} = \widetilde{\rho v_i} v_j + \widetilde{\rho v_j} v_i - v_i v_j \widetilde{\rho}$$

$$\widetilde{v_i} = \frac{1}{\rho} \widetilde{\rho v_i} - \frac{v_i}{\rho} \widetilde{\rho}$$

$$\widetilde{p} = \frac{\gamma - 1}{2} (2\, \widetilde{\rho E} - 2\, \widetilde{\rho v_i} v_i + V^2 \widetilde{\rho})$$

$$\widetilde{T} = \frac{\gamma - 1}{2\rho R} \left[2\, \widetilde{\rho E} - 2\, \widetilde{\rho v_i} v_i + \left(V^2 - \frac{2R}{\gamma - 1} T \right) \widetilde{\rho} \right]$$

$$\widetilde{\rho v_i H} = \widetilde{\rho v_i} H + \gamma v_i \widetilde{\rho E} - (\gamma - 1) v_i \widetilde{\rho v_j} v_j + v_i \left(\frac{\gamma - 1}{2} V^2 - H \right) \widetilde{\rho}$$

with γ the specific heats ratio, R the perfect gas constant and $V^2 = v_i v_i$. Substituting the above relations in eq. 22 and posing

$$h_i = \frac{\partial}{\partial x_j} \left\{ \mu \left[\frac{\partial \lambda_i}{\partial x_j} + \frac{\partial \lambda_j}{\partial x_i} - \frac{2}{3} \frac{\partial \lambda_k}{\partial x_k}\, \delta_{i\,j} \right] \right\}$$

one finds

$$\begin{aligned}
&\int_\Sigma \lambda_i \widetilde{f}_i \, d\Sigma - \int_\Omega \left[v_j \frac{\partial \lambda_i}{\partial x_j} + v_j \frac{\partial \lambda_j}{\partial x_i} - (\gamma-1) v_i \frac{\partial \lambda_j}{\partial x_j} + \frac{h_i}{\rho} \right] \widetilde{\rho v_i} \, d\Omega \\
- &\int_\Omega \left(\frac{\gamma-1}{2} v_j v_j \frac{\partial \lambda_i}{\partial x_i} - v_i v_j \frac{\partial \lambda_i}{\partial x_j} - \frac{1}{\rho} v_i h_i \right) \widetilde{\rho} \, d\Omega \\
- &\int_\Omega (\gamma-1) \frac{\partial \lambda_i}{\partial x_i} \widetilde{\rho E} \, d\Omega + \int_\Sigma \mu \left[\frac{\partial \lambda_i}{\partial x_j} + \frac{\partial \lambda_j}{\partial x_i} - \frac{2}{3} \frac{\partial \lambda_k}{\partial x_k} \delta_{ij} \right] \widetilde{v}_i n_j \, d\Sigma
\end{aligned} \tag{23}$$

The third integral in eq. 20 is rearranged as above to get

$$\begin{aligned}
&\int_\Sigma \zeta e \, d\Sigma - \int_\Sigma k \, \zeta \frac{\partial T}{\partial x_i} \, d\Sigma - \int_\Omega \rho v_i H \frac{\partial \zeta}{\partial x_i} d\Omega - \int_\Sigma k \, T \frac{\partial \zeta}{\partial x_i} n_i \, d\Sigma + \int_\Omega T \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) d\Omega \\
+ &\int_\Omega \mu \frac{\partial \zeta}{\partial x_i} \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{ij} \right] v_j \, d\Omega
\end{aligned} \tag{24}$$

with

$$e = \left\{ \rho v_i H - \mu \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{ij} \right] v_j \right\} n_i$$

The variation of eq. 24 is complicated by the presence of non-linear terms involving both the v_j and the derivatives. As a first step, we write the variation of eq. 24 as

$$\begin{aligned}
&\int_\Sigma \zeta \widetilde{e} \, d\Sigma - \int_\Sigma k \, \zeta \frac{\partial \widetilde{T}}{\partial x_i} n_i \, d\Sigma - \int_\Omega \widetilde{\rho v_i H} \frac{\partial \zeta}{\partial x_i} d\Omega + \int_\Sigma k \, \widetilde{T} \frac{\partial \zeta}{\partial x_i} n_i \, d\Sigma - \int_\Omega \widetilde{T} \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) d\Omega \\
+ &\int_\Omega \left\{ \mu \frac{\partial \zeta}{\partial x_i} \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{ij} \right] - \frac{\partial}{\partial x_i} \left[\mu \left(v_j \frac{\partial \zeta}{\partial x_i} + v_i \frac{\partial \zeta}{\partial x_j} - \frac{2}{3} v_k \frac{\partial \zeta}{\partial x_k} \delta_{ij} \right) \right] \right\} \widetilde{v}_j d\Omega \\
+ &\int_\Sigma \mu \, n_i \left(\frac{\partial \zeta}{\partial x_j} v_i + \frac{\partial \zeta}{\partial x_i} v_j - \frac{2}{3} \frac{\partial \zeta}{\partial x_k} v_k \delta_{ij} \right) \widetilde{v}_j \, d\Sigma
\end{aligned} \tag{25}$$

Leaving unchanged the terms on the boundary and expressing the other variations in terms of the conservative variables variations, the equation above finally becomes

$$\begin{aligned}
&\int_\Sigma \zeta \widetilde{e} \, d\Sigma - \int_\Sigma k \, \zeta \frac{\partial \widetilde{T}}{\partial x_i} \, d\Sigma + \int_\Sigma k \, \widetilde{T} \frac{\partial \zeta}{\partial x_i} n_i \, d\Sigma + \int_\Sigma \mu \, n_i \left(\frac{\partial \zeta}{\partial x_j} v_i + \frac{\partial \zeta}{\partial x_i} v_j - \frac{2}{3} \frac{\partial \zeta}{\partial x_k} v_k \delta_{ij} \right) \widetilde{v}_j \, d\Sigma \\
+ &\int_\Omega \left\{ \left[(\gamma-1) \, v_i v_j - H \delta_{ij} \right] \frac{\partial \zeta}{\partial x_i} + \frac{\gamma-1}{\rho \, R} v_i \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) - \frac{g_j}{\rho} \right\} \widetilde{\rho v_j} \, d\Omega \\
+ &\int_\Omega \left[\frac{g_j v_j}{\rho} - \frac{\gamma-1}{2\rho R} \left(V^2 - \frac{2R}{\gamma-1} T \right) v_j \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) - \left(\frac{\gamma-1}{2} V^2 - H \right) v_i \frac{\partial \zeta}{\partial x_i} \right] \widetilde{\rho} \, d\Omega \\
- &\int_\Omega \left[\frac{\gamma-1}{\rho R} \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) + \gamma v_i \frac{\partial \zeta}{\partial x_i} \right] \widetilde{\rho E} \, d\Omega
\end{aligned} \tag{26}$$

where

$$g_j = \left\{ \frac{\partial}{\partial x_i} \left[\mu \left(v_j \frac{\partial \zeta}{\partial x_i} + v_i \frac{\partial \zeta}{\partial x_j} - \frac{2}{3} v_k \frac{\partial \zeta}{\partial x_k} \delta_{ij} \right) \right] - \mu \frac{\partial \zeta}{\partial x_i} \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{ij} \right] \right\}$$

Note the self-adjointness of the linear part. The variations of the last two integrals in eq. 20 are easily computed as

$$\int_\Sigma \xi_i \widetilde{v_i} \, d\Sigma \tag{27}$$

$$\int_\Sigma \tau \widetilde{\Theta} \, d\Sigma \tag{28}$$

where $\widetilde{\Theta} = \widetilde{T}$ or $\widetilde{\Theta} = \partial \widetilde{T} / \partial x_i n_i$. It is now completed the variation of eq. 20. Consider all the field integrals and factor out the conservative variables variation.

4.3 Adjoint equations and boundary conditions

In order to be

$$\widetilde{L} = 0 \qquad \forall \left\{ \widetilde{\rho}, \widetilde{\rho v_i}, \widetilde{\rho E} \right\}$$

the following equations are to be satisfied

$$-\frac{\gamma-1}{2} v_j v_j \frac{\partial \lambda_i}{\partial x_i} + v_i v_j \frac{\partial \lambda_i}{\partial x_j} - \left(\frac{\gamma-1}{2} V^2 - H \right) v_i \frac{\partial \zeta}{\partial x_i} + \frac{v_j (h_j + g_j)}{\rho} - \left(\frac{\gamma-1}{2\rho R} V^2 - \frac{T}{\rho} \right) \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) = 0 \tag{29}$$

$$\left[(\gamma-1)\, v_i v_j - H \delta_{i\,j} \right] \frac{\partial \zeta}{\partial x_j} - \frac{\partial \eta}{\partial x_i} - v_j \frac{\partial \lambda_i}{\partial x_j} - v_j \frac{\partial \lambda_j}{\partial x_i} + (\gamma-1) v_i \frac{\partial \lambda_j}{\partial x_j} - \frac{h_i + g_i}{\rho} + \frac{\gamma-1}{\rho R} v_i \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) = 0 \tag{30}$$

$$-(\gamma-1) \frac{\partial \lambda_i}{\partial x_i} - \gamma v_i \frac{\partial \zeta}{\partial x_i} - \frac{\gamma-1}{\rho R} \frac{\partial}{\partial x_i} \left(k \frac{\partial \zeta}{\partial x_i} \right) = 0 \tag{31}$$

These are the compressible Navier-Stokes adjoint equations. The boundary conditions to be satisfied by such equations are determined by considering the boundary terms of $\widetilde{L}$. We focus on wing-like geometries, such that in the far field the flow is considered unperturbed, and the no-slip boundary condition is applied on a finite closed surface. In this respect $\widetilde{L}$ depends on the boundary terms left in eqs. 21, 23, 26, 27, 28, once the field integrals are elided by satisfying the adjoint equations. We are left with

$$\begin{aligned} \widetilde{L} = \widetilde{I} + \int_\Sigma \lambda_i \widetilde{f}_i \, d\Sigma + \int_\Sigma \eta \, \widetilde{\rho} v_i n_i \, d\Sigma + \int_\Sigma \zeta \widetilde{e} \, d\Sigma - k \int_\Sigma \zeta \frac{\partial \widetilde{T}}{\partial x_i} n_i \, d\Sigma + k \int_\Sigma \widetilde{T} \frac{\partial \zeta}{\partial x_i} n_i \, d\Sigma + \int_\Sigma \tau \widetilde{\Theta} d\Sigma \\ + \int_\Sigma \left[\xi_i + \eta \rho n_i + \mu \left(\frac{\partial \lambda_i}{\partial x_j} n_j + \frac{\partial \lambda_j}{\partial x_i} n_j - \frac{2}{3} \frac{\partial \lambda_j}{\partial x_j} n_i + n_j \frac{\partial \zeta}{\partial x_i} v_j + n_j \frac{\partial \zeta}{\partial x_j} v_i - \frac{2}{3} n_i \frac{\partial \zeta}{\partial x_j} v_j \right) \right] \widetilde{v}_i \, d\Sigma \end{aligned} \tag{32}$$

Suppose we give a certain wall temperature. Then $\widetilde{\Theta} = \widetilde{T}$, and therefore by taking

$$\zeta = 0 \text{ and } \tau = -k \frac{\partial \zeta}{\partial x_i} n_i \qquad \textit{Prescribed wall temperature} \tag{33}$$

on Σ, the boundary integrals involving $\widetilde{T} = 0$ and $\partial \widetilde{T} \partial x_i$ are eliminated. In contrast, the adiabatic wall corresponds to $\widetilde{\Theta} = (\partial \widetilde{T} / \partial x_i) n_i$, and hence if

$$\frac{\partial \zeta}{\partial x_i} n_i = 0 \text{ and } \tau = k\zeta \qquad \textit{Adiabatic wall} \tag{34}$$

on Σ, the respective boundary integrals are eliminated as in the previous case. As $v_i = 0$ on the solid walls, we have remarkable simplifications for $\widetilde{e}$:

$$\widetilde{e} = \left\{ \rho H n_i - \mu \left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{i\,j} \right] n_j \right\} \widetilde{v}_i$$

Also the third term in 32 is 0 and if we take

$$\xi_i = -\left[\rho H n_i - \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \frac{\partial v_k}{\partial x_k} \delta_{i\,j} \right) n_j \right] \zeta - \mu \left(\frac{\partial \lambda_i}{\partial x_j} + \frac{\partial \lambda_j}{\partial x_i} - \frac{2}{3} \frac{\partial \lambda_k}{\partial x_k} \delta_{i\,j} \right) n_j - \rho n_i \eta \tag{35}$$

we are left with

$$\widetilde{L} = \widetilde{I} + \int_\Sigma \lambda_i \widetilde{f_i}\, d\Sigma \tag{36}$$

where $\widetilde{f}$ is the variation of the force per unit surface acting on the fluid at the boundary.

Such variation is unconstrained, and there are two different ways to elide the two terms left in the equation above. The first is that $\widetilde{I}$ can be put under the form

$$\widetilde{I} = \int_\Sigma J_i \widetilde{f_i}\, d\Sigma \tag{37}$$

so that taking

$$\lambda_i = -J_i \tag{38}$$

the variation of the lagrangian is finally zero. In turn this condition shows how the only allowable surface cost functionals are those involving f_i, as other functionals will not lead to $\widetilde{L} = 0$. For a deeper discussion see [11].

Otherwise, if I is a field integral then

$$\widetilde{I} = \int_\Omega J\{\widetilde{\rho}, \widetilde{\rho v_i}, \widetilde{\rho E}\}\widetilde{f}\, d\Omega \tag{39}$$

where in this case J is the derivative of the integrand with respect to the conservative variables. The three terms under integral become source terms of the adjoint equations, and by taking $\lambda_i = 0$ on Σ we are left again with $\widetilde{L} = 0$.

4.4 Gradient

As mentioned, the variation of L with respect to the Lagrange multipliers yields back the governing equations and boundary conditions, therefore the gradient is simply the variation with respect to the geometry. If I is a field integral, and if $\epsilon\,\omega$ is variation in the direction of the normal to the boundary, we have

$$\mathcal{D}I = \int_\Omega \frac{\partial F}{\partial \omega}\omega\, d\Omega + \int_\Sigma F\omega\, d\Sigma \tag{40}$$

where F is the integrand of I and ϵ is small. When I is defined on the boundary we have

$$\mathcal{D}I = \int_\Sigma \frac{\partial F}{\partial \omega}\omega\, d\Sigma + \int_\Sigma \frac{\partial F}{\partial x_i} n_i \omega\, d\Sigma + \int_\Sigma HF\,\omega\, d\Sigma \tag{41}$$

and H is the local surface curvature.

For the other terms of L the derivation rules are the same, but some simplifications occur since the governing and adjoint equations with the respective boundary conditions are satisfied on Σ. The only terms left are those relative to the boundary terms and the gradient, that is

$$\mathcal{D}L = \mathcal{D}I + \int_\Sigma \frac{\partial \xi_i v_i}{\partial x_j} n_j \omega\, d\Sigma + \int_\Sigma \frac{\partial \tau\Theta}{\partial x_i} n_i \omega\, d\Sigma \tag{42}$$

In the particular case under examination the functional to be minimized is

$$\mathcal{L} = \omega_1 D + \omega_2 \frac{(L - L^*)^2}{2} \tag{43}$$

where D is the drag (to be minimized), L is the lift, L^* is the desired lift and the ω_i are weights. Consequently, the gradient expression is:

$$\begin{aligned}
\delta\mathcal{L} \;=\; & \omega_1 \int_\Sigma -\left(-\tfrac{\partial p\, n_i}{\partial y}\delta y + \tfrac{\partial \tau_{ij}\, n_j}{\partial y}\delta y\right) t_i^\infty\, d\Sigma + \\
& \omega_2 \left(L - L^*\right) \int_\Sigma -\left(-\tfrac{\partial p\, n_i}{\partial y}\delta y + \tfrac{\partial \tau_{ij}\, n_j}{\partial y}\delta y\right) n_i^\infty\, d\Sigma + \\
& \int_\Sigma \xi_i \tfrac{\partial v_i}{\partial y}\delta y\, d\Sigma + \int_\Sigma \tau \tfrac{\partial \Theta}{\partial y}\delta y\, d\Sigma
\end{aligned}$$

5 Use of approximate fitness evaluators

Optimization techniques based on evolutionary computing are very attractive in terms of robustness and global extremum location capabilities, but these favorable characteristics are obtained to the expense of a rather large number of fitness evaluations. This is a problem in many engineering applications, where fitness evaluation often requires a substantial amount of computational resources. This makes attractive the use of approximate fitness evaluators, with lower computational requirements, whenever it is possible. In aerodynamic optimization design, for example, various approaches have been experimented, such as the use of Neural Network-based interpolators to reduce the number of true flow field evaluations [12]. The mixed use of solvers with different levels of approximation in a hierarchical organization has also been object of investigation. Both procedures have advantages and limits. Neural Networks, for example, are able to improve the quality of their approximation when the number of exact evaluations available increase. Their performance, therefore, may improve in real time during the optimization process. On the other hand, their more serious drawback is the fast decrease in approximation fidelity when the parameter space dimension increases [13, 14, 15, 16]. Low fidelity solvers (e.g. Euler+boundary layer versus Navier-Stokes), do not have this problem, but it is not always easy to understand the limits of the approximate model. In both cases, the problem that arises is the difficulty of devising a good and sound strategy to mix and interleave high-fidelity and low-fidelity fitness approximations.

5.1 Problem analysis

A simple analysis of the implications of mixing fitness evaluators with different levels of precision within an (evolutionary) optimization process is here carried out. This analysis is made using concepts of multi-objective optimization [17, 18, 19], like Pareto fronts and dominance criteria, even if the class of problems considered is strictly single-objective. In fact, the relations between the exact function f and its approximated model g is analyzed with the following two-objective problem:

$$\begin{cases} \min f(\mathbf{x}),\ g(\mathbf{x}, X_0) \\ \mathbf{x} \in X \\ X_0 \in \wp(X) \end{cases} \tag{44}$$

where $f(\mathbf{x}) \in \Re$ is the function defined in a domain $X \subseteq \Re^n$ for which we are interested in finding the global extrema, and $g(\mathbf{x}, X_0) \in \Re$ is defined in the domain $X^* \equiv X \cup \wp(X)$; $\wp(X)$ is the power set of X and $X_0 \in \wp(X)$ represents a sampling of the domain X. In other terms X_0 can be thought as the training set of g, e.g. the set of point in which f (and possibly its derivatives) are known and their values are used to obtain g. Let us have, for example, the following function:

$$f(x) = [x^2 - 10\cos(2\pi x) + 10] \quad x \in [-1, 3] \tag{45}$$

and let $g(x)$ a polynomial of degree 10 that minimizes $\sum_{x_i \in X_0} [f(x_i) - g(x_i)]^2$ to best fit $f(x)$ in the least squares sense for a given training set X_0 (see Figure 2).

The Pareto Front related to problem (44), reported here in Figure 3 for function 45 and its approximator, is useful to understand the relationship between f and its approximating function g.

The training set X_0 is, for the moment, supposed fixed and unchanged during the optimization process. In other words, X_0 fixed means that no learning procedure is involved in the approximation process; there are, instead, two different models, from the beginning of the process, the exact and the approximate one, that can be used to search the optimum.

When using an approximated model for fitness evaluation, there are two things that have to be taken into account:

- Shift of extremum points between the two models;
- Value shift between exact and approximate functions.

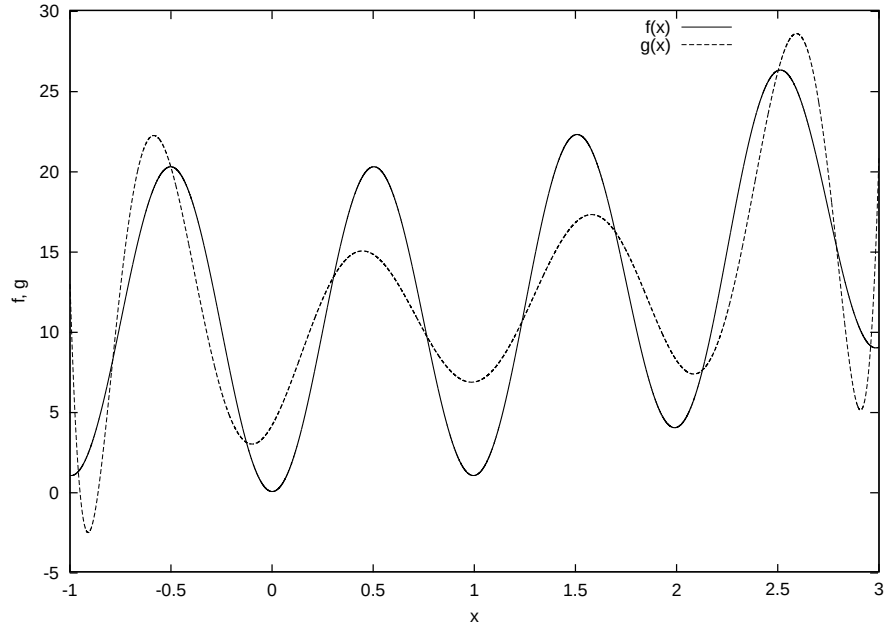

Figure 2: $f(x) = [x^2 - 10\cos(2\pi x) + 10]$ and g = best fitting polynomial of degree 10.

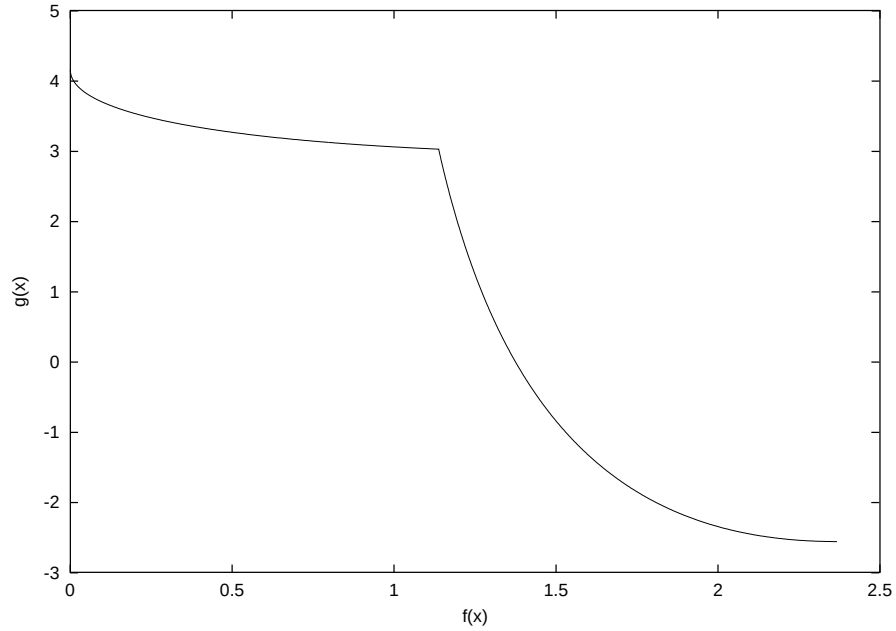

Figure 3: Pareto front of problem 44 with f specified in eq. 45 and g 10^{th} degree polynomial approximator.

The lesser are these shifts in the extremum points, the better and more reliable are optimization results obtained using the approximate model. Furthermore in these conditions it is safe to interleave exact and approximate evaluations.

A generic Pareto front of a problem of type (1) is reported in Figure 4. Four different zones can be found, that are characterized by different behavior of g and f in moving towards their global optimum points. In zone (I), a solution that dominates the initial one is surely closer to both f and g global minimum points. In zone (II), a solution that dominates the initial one is closer to the global minimum of f and farther from g global minimum. Conversely, zone (IV) has an opposite behavior, as a dominating solution leads close to g and far from f global minimum points. In zone (III), finally, a solution that dominates the initial one is farther from both global minimum points.

Given this framework, we want check if there is a particular interleaving strategy that minimizes the number of exact function evaluations. The obvious limit of this approach is that in a real optimization run the Pareto front is unknown and only a rough estimate of the collocation of a particular solution in one of the four zones is possible.

At the beginning of the optimization process the candidate solution (or population) is likely to be in zone (I). The approximated function g can be safely used here, and only occasional control is needed on f values to avoid the generation of too many non dominated solutions (relatively to f).

Zones (II) and (III) are difficult to detect if g global minimum value is unknown. In principle, use of g without check on f should be avoided in zone (II), as it may lead to unsatisfactory results in terms of f. In zone (III) both the use of g alone and of g and f interleaved is detrimental, as it would lead to a point in the Pareto front that would be likely to be far from the global optimum of f.

Zone (IV) is similar to zone (I) from the point of view of searching for the global optimum of f. However, here interleaving could be detrimental for reaching the global optimum of g.

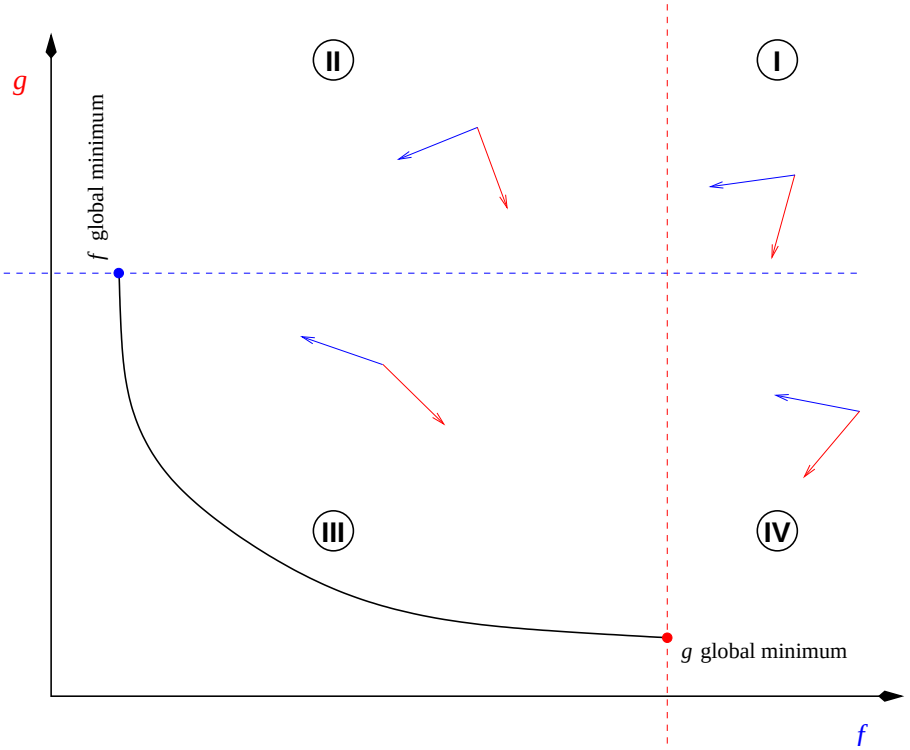

Figure 4: Generic Pareto front related to problem 44.

5.2 Possible approaches to mixing exact and approximated evaluations

From a practical standpoint the above considerations suggest the following search strategy:

1. Search the global optimum of g first, eventually checking in very few points if the values of f do not get worse too much.

2. Search the global optimum of f, eventually using g as second objective if there is interest in finding solutions belonging to the f-g front.

When X_0 changes during the optimization process, the approximation g can increase its precision when more evaluation of f are available. Therefore the Pareto front tends to decrease in size, and after some time it can reduce to a point.

Devising a good interleaving strategy is here more difficult, however the previous strategy, adapted as follows, should have an acceptably good behavior:

1. f is sampled in some points and the resulting values (training set) are used to find a first expression of g.
2. The global optimum of g is searched.
3. A short two-objective (f-g) run is performed and the computed values of f are added to the training set.
4. A new expression of g is found.
5. The process is repeated starting from step 2 until a satisfactory value of f is found.

5.3 Analysis of an example problem

The RAE 2822 airfoil is assigned as starting configuration, and the optimization goal is the reduction of the drag coefficient c_d at given mach ($M = 0.78$) and lift coefficient $c_l = 0.60$. The maximum airfoil thickness has to remain unchanged.

A two-objective optimization run is carried out using two different flow solvers:

1. Euler + Boundary layer, interactive (high fidelity solver, objective f).
2. Full potential, non-conservative formulation (low fidelity solver, objective g).

It is important to observe that these flow solvers represent the real flow at a very different approximation level, and that a real optimization application would benefit more from the use of more closely related flow solver models, such as Navier-Stokes and Euler + Boundary layer. The only aim of this run is the analysis of the relationships between different flow models.

The aerofunctions obtained from the airfoils reported in figure 5 were used to modify the airfoil shape.

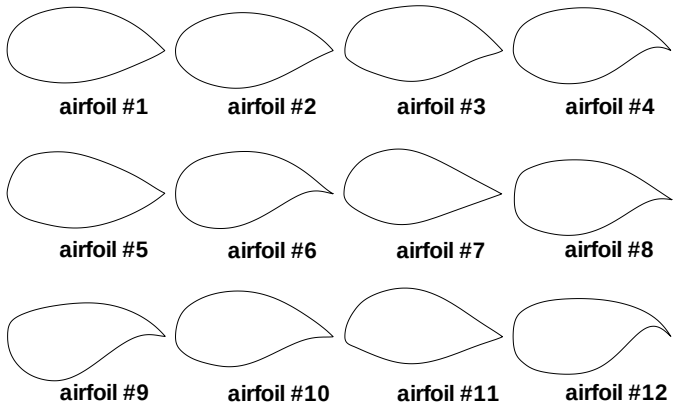

Figure 5: Airfoils used to build the modification functions.

The GA parameters are:

- bit resolution: 10
- selection method: two-step random walk with elitism
- bit-mutation with 4% rate
- population size: 20
- generations: 20

Figure 6 reports the Pareto front obtained in the two-objective run. As it can be observed, the Pareto front shows a sensible difference between the two flow models. These solvers should, therefore, not be used in an interleaved fashion unless a viscous correction is introduced in the full potential one. However, a hierarchical use of these solver could be useful to discard very bad solution such as those with a strong shock wave.

The above example may help in tracing the guidelines for the implementation of the hierarchical approach in an optimizer for viscous flows.

The optimization could be made in two different steps: a pre-optimization phase aimed at the reduction of the search space complexity, and a subsequent optimization phase that explores this reduced space using a high fidelity fitness evaluator.

The number of active variables, their variation ranges and a new starting configuration are a possible output of the first phase. In a more sophisticated approach, using for example a multi-objective GA, the pre-optimization can be used to generate an aerofunction basis with a reduced number of elements.

In the second phase, the modification of the geometry could be driven by a very simple evolution strategy, such as a (1,1)-ES [20, 21] that, due reduction of the search space complexity, should give an improvement of the objective function in an acceptable number of attempts.

6 Conclusions

It has been shown how computer algebra techniques can be used to obtain the adjoint equations and pertinent boundary conditions of the Navier-Stokes equations. The obtained equations can be directly used for the adjoint equations solver coding.

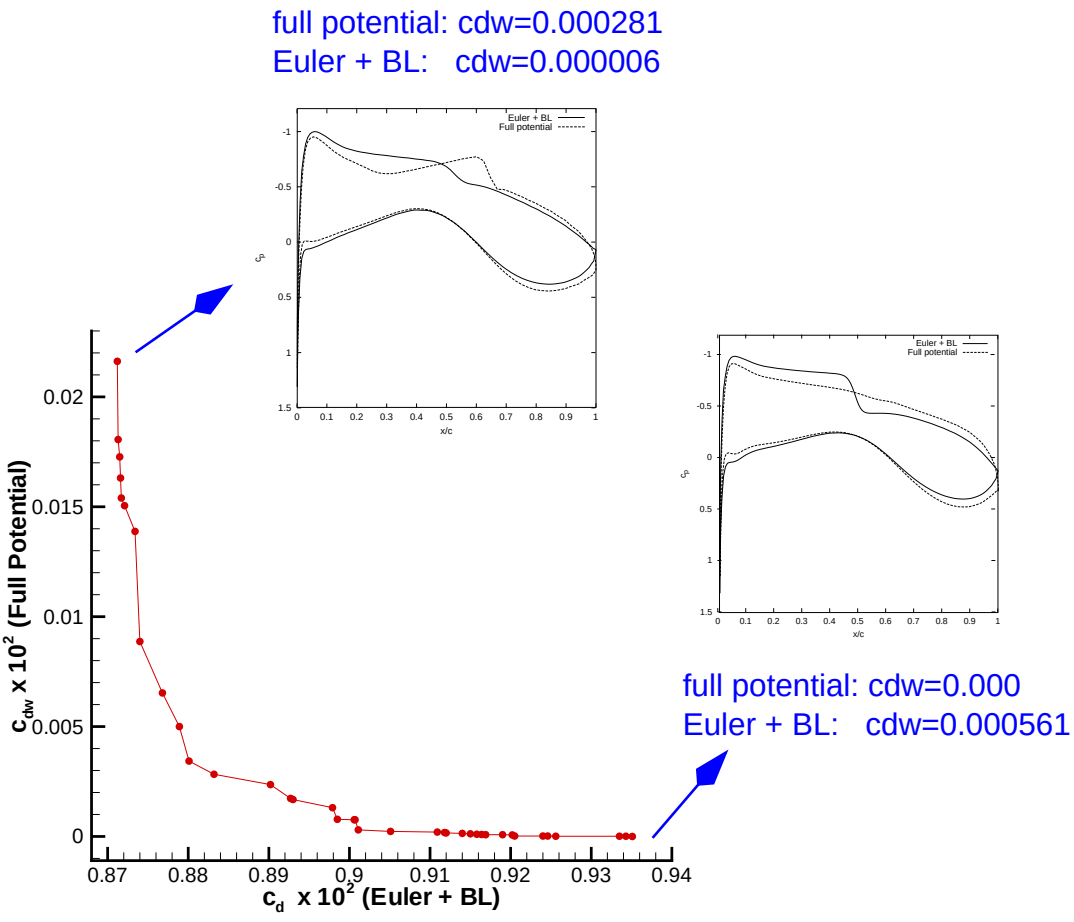

Figure 6: Euler + Boundary Layer — Full Potential comparison.

MAPLE was used as computer algebra language, complemented with an original code taking advantage of public domain libraries for tensor algebra and function variation computation.

Further development will include derivation of the adjoint equations of the Reynolds averaged Navier-Stokes equations with κ-ϵ turbulence modeling.

The relationship between an objective function and an approximated form has been investigated using a Pareto front to show the trade-off between exact and expensive objective function evaluations and inexpensive but inexact approximations. Approximating the objective function values, if properly done, shortens the time in time-consuming design problems like those typical of CFD applications. Evolutionary algorithms normally require more objective function evaluations than other methods and, therefore, are very likely to take advantage by using such approximation techniques.

Acknowledgments

Part of the work here presented was carried out within the framework of the AEROSHAPE project. This project (Multi-Point Aerodynamic Shape Optimization) is a collaboration between Aerospatiale Matra Airbus, Alenia Aeronautica (Coordinator), Daimler Crysler Airbus, EADS-M, Dassault Aviation, SAAB, SENER, SYNAPS, CIRA, DERA, DLR, FFA, INRIA, HCSA, NLR, ONERA, and NuTech Solutions. The project is funded by the European Commission, DG Research, under the GROWTH initiative (Project Ref.: GRD1-1999-10752).

A Example of adjoint NS derivation using MAPLE

For the sake of clarity and conciseness, the output of MAPLE has been slightly modified such that indexed variables and derivatived such as $x1$ and $v3_{x2}$ appear as x_1 and v_{3x_2}. Furthermore, we will work only on a piece of the functional 20, namely the scalar products between the adjoint variables and the state equations. Therefore, in the output will not appear the terms coming from I. Hence we have the following expanded form:

$$
\begin{aligned}
LG1 := \iiint & \eta\,(\rho_{x_1}\,v_1 + \rho\,v_{1\,x_1} + \rho_{x_2}\,v_2 + \rho\,v_{2\,x_2} + \rho_{x_3}\,v_3 + \rho\,v_{3\,x_3}) + \lambda_1(\rho_{x_1}\,{v_1}^2 \\
& + 2\,\rho\,v_1\,v_{1\,x_1} + p_{x_1} - \frac{4}{3}\,\mu\,v_{1\,x_1,\,x_1} - \frac{1}{3}\,\mu\,v_{2\,x_1,\,x_2} - \frac{1}{3}\,\mu\,v_{3\,x_1,\,x_3} + \rho_{x_2}\,v_2\,v_1 + \rho\,v_{2\,x_2}\,v_1 \\
& + \rho\,v_2\,v_{1\,x_2} - \mu\,v_{1\,x_2,\,x_2} + \rho_{x_3}\,v_3\,v_1 + \rho\,v_{3\,x_3}\,v_1 + \rho\,v_3\,v_{1\,x_3} - \mu\,v_{1\,x_3,\,x_3}) + \lambda_2 \\
& (\rho_{x_1}\,v_2\,v_1 + \rho\,v_{2\,x_1}\,v_1 + \rho\,v_2\,v_{1\,x_1} - \mu\,v_{2\,x_1,\,x_1} - \frac{1}{3}\,\mu\,v_{1\,x_1,\,x_2} + \rho_{x_2}\,{v_2}^2 + 2\,\rho\,v_2\,v_{2\,x_2} \\
& + p_{x_2} - \frac{4}{3}\,\mu\,v_{2\,x_2,\,x_2} - \frac{1}{3}\,\mu\,v_{3\,x_2,\,x_3} + \rho_{x_3}\,v_3\,v_2 + \rho\,v_{3\,x_3}\,v_2 + \rho\,v_3\,v_{2\,x_3} - \mu\,v_{2\,x_3,\,x_3}) + \\
& \lambda_3(\rho_{x_1}\,v_3\,v_1 + \rho\,v_{3\,x_1}\,v_1 + \rho\,v_3\,v_{1\,x_1} - \mu\,v_{3\,x_1,\,x_1} - \frac{1}{3}\,\mu\,v_{1\,x_1,\,x_3} + \rho_{x_2}\,v_3\,v_2 \\
& + \rho\,v_{3\,x_2}\,v_2 + \rho\,v_3\,v_{2\,x_2} - \mu\,v_{3\,x_2,\,x_2} - \frac{1}{3}\,\mu\,v_{2\,x_2,\,x_3} + \rho_{x_3}\,{v_3}^2 + 2\,\rho\,v_3\,v_{3\,x_3} + p_{x_3} \\
& - \frac{4}{3}\,\mu\,v_{3\,x_3,\,x_3}) + \zeta(-\frac{1}{3}\,v_3\,\mu\,v_{1\,x_1,\,x_3} - \frac{4}{3}\,{v_{3\,x_3}}^2\,\mu - {v_{2\,x_3}}^2\,\mu + \rho_{x_3}\,v_3\,H + \rho\,v_{3\,x_3}\,H \\
& + \rho\,v_3\,H_{x_3} - k\,T_{x_3,\,x_3} - {v_{1\,x_3}}^2\,\mu - k\,T_{x_1,\,x_1} - \frac{1}{3}\,v_2\,\mu\,v_{1\,x_1,\,x_2} - k\,T_{x_2,\,x_2} - \frac{1}{3}\,v_2\,\mu\,v_{3\,x_2,\,x_3} \\
& - v_1\,\mu\,v_{1\,x_2,\,x_2} - \frac{4}{3}\,v_2\,\mu\,v_{2\,x_2,\,x_2} - v_3\,\mu\,v_{3\,x_1,\,x_1} - \frac{1}{3}\,v_1\,\mu\,v_{2\,x_1,\,x_2} - \frac{4}{3}\,v_3\,\mu\,v_{3\,x_3,\,x_3} \\
& - \frac{1}{3}\,v_3\,\mu\,v_{2\,x_2,\,x_3} - v_1\,\mu\,v_{1\,x_3,\,x_3} - v_3\,\mu\,v3_{x_2,\,x_2} - \frac{4}{3}\,{v_{1\,x_1}}^2\,\mu + \rho\,v_1\,H_{x_1} + \rho\,v_{1\,x_1}\,H \\
& + \rho_{x_1}\,v_1\,H - {v_{3\,x_1}}^2\,\mu - v_2\,\mu\,v_{2\,x_3,\,x_3} - 2\,v_{2\,x_1}\,\mu\,v_{1\,x_2} - {v_{2\,x_1}}^2\,\mu + \frac{4}{3}\,v_{1\,x_1}\,\mu\,v_{3\,x_3} \\
& + \frac{4}{3}\,v_{1\,x_1}\,\mu\,v_{2\,x_2} - v_2\,\mu\,v_{2\,x_1,\,x_1} - 2\,v_{3\,x_1}\,\mu\,v_{1\,x_3} - \frac{4}{3}\,v_1\,\mu\,v_{1\,x_1,\,x_1} - \frac{1}{3}\,v_1\,\mu\,v_{3\,x_1,\,x_3} \\
& - \frac{4}{3}\,{v_{2\,x_2}}^2\,\mu - {v_{1\,x_2}}^2\,\mu + \rho\,v_2\,H_{x_2} + \rho\,v_{2\,x_2}\,H + \rho_{x_2}\,v_2\,H - {v_{3\,x_2}}^2\,\mu + \frac{4}{3}\,v_{2\,x_2}\,\mu\,v_{3\,x_3} \\
& - 2\,v_{3\,x_2}\,\mu\,v_{2\,x_3})dx_1 dx_2 dx_3 + \iint \xi_1\,v_1 + \xi_2\,v_2 + \xi_3\,v_3 + \tau\,\Theta(T)\,ds_1\,ds_2
\end{aligned}
$$

Now the perfect gas model is introduced in order to close the equation system:

```
> pexp := p = -1/2*rho*(gamma-1)*(-2*H+v1^2+v2^2+v3^2)/gamma;
```

$$pexp := p = -\frac{1}{2}\,\frac{\rho\,(\gamma-1)\,(-2\,H + {v_1}^2 + {v_2}^2 + {v_3}^2)}{\gamma}$$

```
> Texp := T = -1/2*(gamma-1)*(-2*H+v1^2+v2^2+v3^2)/(gamma*R);
```

$$Texp := T = -\frac{1}{2}\,\frac{(\gamma-1)\,(-2\,H + {v_1}^2 + {v_2}^2 + {v_3}^2)}{\gamma\,R}$$

```
> LG1:=subs({pexp,Texp}, LG1):
```

The prescribed boundary temperature condition is imposed:

```
> Theta := (w) -> w- Tw(x1,x2,x3,s1,s2);
```

$$\Theta := w \to w - \mathrm{Tw}(x1,\ x2,\ x3,\ s1,\ s2)$$

The variation with respect to ρ gives the first adjoint equation:

```
> adj_drho:=compute_adj_eq(LG1, rho, drho, 2, 0, true);
```

$$-v_2\,H\,\zeta_{x_2}-\frac{1}{2}\,\frac{{v_3}^2\,\lambda_{3\,x_3}}{\gamma}-v_2\,v_3\,\lambda_{3\,x_2}-v_2\,v_3\,\lambda_{2\,x_3}+\frac{H\,\lambda_{3\,x_3}}{\gamma}+\frac{H\,\lambda_{2\,x_2}}{\gamma}-\eta_{x_3}\,v_3+\frac{\lambda_{1\,x_1}\,H}{\gamma}$$
$$-\lambda_{1\,x_1}\,H-\eta_{x_1}\,v_1-H\,\lambda_{3\,x_3}-\eta_{x_2}\,v_2+\frac{1}{2}\,{v_1}^2\,\lambda_{3\,x_3}-\frac{1}{2}\,{v_1}^2\,\lambda_{1\,x_1}+\frac{1}{2}\,{v_1}^2\,\lambda_{2\,x_2}$$
$$+\frac{1}{2}\,{v_3}^2\,\lambda_{2\,x_2}+\frac{1}{2}\,\lambda_{1\,x_1}\,{v_3}^2+\frac{1}{2}\,\lambda_{1\,x_1}\,{v_2}^2+\frac{1}{2}\,{v_2}^2\,\lambda_{3\,x_3}-\frac{1}{2}\,{v_2}^2\,\lambda_{2\,x_2}-\frac{1}{2}\,{v_3}^2\,\lambda_{3\,x_3}-H\,\lambda_{2\,x_2}$$
$$-v_1\,H\,\zeta_{x_1}-\frac{1}{2}\,\frac{\lambda_{1\,x_1}\,{v_2}^2}{\gamma}-v_1\,v_3\,\lambda_{1\,x_3}-\frac{1}{2}\,\frac{{v_1}^2\,\lambda_{3\,x_3}}{\gamma}-\frac{1}{2}\,\frac{{v_3}^2\,\lambda_{2\,x_2}}{\gamma}-v_1\,v_3\,\lambda_{3\,x_1}$$
$$-\frac{1}{2}\,\frac{{v_1}^2\,\lambda_{2\,x_2}}{\gamma}-\frac{1}{2}\,\frac{{v_1}^2\,\lambda_{1\,x_1}}{\gamma}-\zeta_{x_3}\,v_3\,H-\frac{1}{2}\,\frac{{v_2}^2\,\lambda_{3\,x_3}}{\gamma}-v_1\,v_2\,\lambda_{1\,x_2}-v_1\,v_2\,\lambda_{2\,x_1}$$
$$-\frac{1}{2}\,\frac{\lambda_{1\,x_1}\,{v_3}^2}{\gamma}-\frac{1}{2}\,\frac{{v_2}^2\,\lambda_{2\,x_2}}{\gamma}$$

Now the adjoint equations coming from the variations with respect to v_1, v_2, v_3:

```
> adj_dv1:=compute_adj_eq(LG1, v1, dv1, 2, 0, true);
```

$$-\rho\,H\,\zeta_{x_1}+\frac{5}{3}\,\mu\,v_{2\,x_1}\,\zeta_{x_2}-\mu\,\zeta_{x_3,\,x_3}\,v_1-\frac{4}{3}\,\mu\,v_1\,\zeta_{x_1,\,x_1}-\mu\,\lambda_{1\,x2,\,x_2}-\rho\,v_3\,\lambda_{1\,x_3}$$
$$+\rho\,v_1\,\lambda_{2\,x_2}-\rho\,v_3\,\lambda_{3\,x_1}-\frac{\rho\,v_1\,\lambda_{1\,x_1}}{\gamma}-\frac{\rho\,v_1\,\lambda_{2\,x_2}}{\gamma}-\frac{\rho\,v_1\,\lambda_{3\,x_3}}{\gamma}+\frac{k\,v_1\,\zeta_{x_1,\,x_1}}{R}$$
$$+\frac{k\,v_1\,\zeta_{x_2,\,x_2}}{R}+\frac{k\,\zeta_{x_3,\,x_3}\,v_1}{R}-\eta_{x1}\,\rho-\frac{5}{3}\,\mu\,v_{3\,x_3}\,\zeta_{x_1}-\frac{5}{3}\,\mu\,\zeta_{x_1}\,v_{2\,x_2}$$
$$-\frac{1}{3}\,\mu\,v_3\,\zeta_{x_1,\,x_3}+\frac{5}{3}\,\mu\,v_{3\,x_1}\,\zeta_{x_3}-\frac{4}{3}\,\mu\,\lambda_{1\,x_1,\,x_1}+\rho\,v_1\,\lambda_{3\,x_3}-\frac{1}{3}\,\mu\,\lambda_{2\,x_1,\,x_2}-\rho\,v_2\,\lambda_{1\,x2}$$
$$-\frac{1}{3}\,\mu\,v_2\,\zeta_{x_1,\,x_2}-\frac{1}{3}\,\mu\,\lambda_{3\,x_1,\,x_3}-\rho\,v_1\,\lambda_{1\,x_1}-\mu\,v_1\,yH_{x_2,\,x_2}-\mu\,\lambda_{1\,x_3,\,x_3}-\frac{k\,\zeta_{x_3,\,x_3}\,v_1}{\gamma\,R}$$
$$-\frac{k\,v_1\,\zeta_{x_1,\,x_1}}{\gamma\,R}-\frac{k\,v_1\,\zeta_{x_2,\,x_2}}{\gamma\,R}-\rho\,v_2\,\lambda_{2\,x_1}$$

```
> adj_dv2:=compute_adj_eq(LG1, v2, dv2, 2, 0, true);
```

$$-\mu\,\zeta_{x_1,\,x_1}\,v_2-\frac{1}{3}\,\mu\,v_3\,\zeta_{x_2,\,x_3}+\rho\,v_2\,\lambda_{3\,x_3}+\rho\,v_2\,\lambda_{1\,x_1}+\frac{5}{3}\,\mu\,v_{3\,x_2}\,\zeta_{x_3}+\frac{5}{3}\,\mu\,v_{1\,x_2}\,\zeta_{x_1}$$
$$-\mu\,\lambda_{2\,x_1,\,x_1}-\frac{\rho\,v_2\,\lambda_{1\,x_1}}{\gamma}-\frac{\rho\,v_2\,\lambda_{3\,x_3}}{\gamma}-\frac{\rho\,v_2\,\lambda_{2\,x_2}}{\gamma}-\eta_{x_2}\,\rho-\frac{4}{3}\,\mu\,\lambda_{2\,x_2,\,x_2}$$
$$-\frac{k\,\zeta_{x_1,\,x_1}\,v_2}{\gamma\,R}-\frac{k\,v_2\,\zeta_{x_3,\,x_3}}{\gamma\,R}-\frac{k\,v_2\,\zeta_{x_2,\,x_2}}{\gamma\,R}-\frac{1}{3}\,\mu\,\lambda_{3\,x_2,\,x_3}-\frac{4}{3}\,\mu\,v_2\,\zeta_{x_2,\,x_2}$$
$$-\rho\,v_3\,\lambda_{2\,x_3}-\frac{5}{3}\,\mu\,\zeta_{x_2}\,v_{1\,x_1}-\rho\,v_2\,\lambda_{2\,x_2}-\mu\,v_2\,\zeta_{x_3,\,x_3}-\rho\,H\,\zeta_{x_2}-\frac{1}{3}\,\mu\,v_1\,\zeta_{x_1,\,x_2}$$
$$-\rho\,v_3\,\lambda_{3\,x_2}-\rho\,v_1\,\lambda_{2\,x_1}-\frac{1}{3}\,\mu\,\lambda_{1\,x_1,\,x_2}-\rho\,v_1\,\lambda_{1\,x_2}+\frac{k\,v_2\,\zeta_{x_3,\,x_3}}{R}+\frac{k\,v_2\,\zeta_{x_2,\,x_2}}{R}$$
$$+\frac{k\,\zeta_{x_1,\,x_1}\,v_2}{R}-\mu\,\lambda_{2\,x_3,\,x_3}-\frac{5}{3}\,\mu\,v_{3\,x_3}\,\zeta_{x_2}$$

```
> adj_dv3:=compute_adj_eq(LG1, v3, dv3, 2, 0, true);
```

$$-\frac{1}{3}\,\mu\,\zeta_{x_1,\,x_3}\,v_1-\frac{1}{3}\,\mu\,\lambda_{2\,x_2,\,x_3}-\rho\,v_2\,\lambda_{2\,x_3}-\rho\,v_3\,\lambda_{3\,x_3}-\rho\,v_1\,\lambda_{1\,x3}-\rho\,H\,\zeta_{x_3}-\frac{1}{3}\,\mu\,\lambda_{1\,x_1,\,x_3}$$
$$-\mu\,\lambda_{3\,x_1,\,x_1}-\mu\,\lambda_{3\,x_2,\,x_2}-\eta_{x_3}\,\rho-\frac{\rho\,v_3\,\lambda_{2\,x_2}}{\gamma}-\frac{\rho\,v_3\,\lambda_{1\,x_1}}{\gamma}-\frac{\rho\,v_3\,\lambda_{3\,x_3}}{\gamma}$$
$$-\frac{4}{3}\,\mu\,\lambda_{3\,x_3,\,x_3}-\frac{k\,\zeta_{x_2,\,x_2}\,v_3}{\gamma\,R}-\frac{k\,\zeta_{x_1,\,x_1}\,v_3}{\gamma\,R}+\frac{k\,v_3\,\zeta_{x_3,\,x_3}}{R}+\frac{k\,\zeta_{x_1,\,x_1}\,v_3}{R}$$
$$+\frac{k\,\zeta_{x_2,\,x_2}\,v_3}{R}-\frac{k\,v_3\,\zeta_{x_3,\,x_3}}{\gamma\,R}-\rho\,v_2\,\lambda_{3\,x_2}+\frac{5}{3}\,\mu\,v_{1\,x_3}\,\zeta_{x_1}-\frac{5}{3}\,\mu\,\zeta_{x_3}\,v_{1\,x_1}$$
$$-\rho\,v_1\,\lambda_{3\,x_1}+\rho\,v_3\,\lambda_{1\,x_1}-\frac{5}{3}\,\mu\,\zeta_{x_3}\,v_{2\,x_2}+\frac{5}{3}\,\mu\,\zeta_{x_2}\,v_{2\,x_3}+\rho\,v_3\,\lambda_{2\,x_2}-\mu\,\zeta_{x_2,\,x_2}\,v_3$$
$$-\frac{4}{3}\,\mu\,v_3\,\zeta_{x_3,\,x_3}-\mu\,\zeta_{x_1,\,x_1}\,v_3-\frac{1}{3}\,\mu\,v_2\,\zeta_{x_2,\,x_3}$$

Finally the variation with respect to H gives:

```
> adj_dH:=compute_adj_eq(LG1, H, dH, 2, 0, true);
```

$$-\rho\,\lambda_{3\,x_3}+\frac{k\,\zeta_{x_1,\,x_1}}{\gamma\,R}-\rho\,v_2\,\zeta_{x_2}-\rho\,\lambda_{2\,x_2}-\rho\,v_1\,\zeta_{x_1}-\rho\,v_3\,\zeta_{x_3}+\frac{k\,\zeta_{x_3,\,x_3}}{\gamma\,R}+\frac{k\,\zeta_{x_2,\,x_2}}{\gamma\,R}$$
$$+\frac{\rho\,\lambda_{1\,x_1}}{\gamma}-\frac{k\,\zeta_{x_1,\,x_1}}{R}-\frac{k\,\zeta_{x_3,\,x_3}}{R}+\frac{\rho\,\lambda_{2\,x_2}}{\gamma}-\frac{k\,\zeta_{x_2,\,x_2}}{R}-\rho\,\lambda_{1\,x_1}+\frac{\rho\,\lambda_{3\,x_3}}{\gamma}$$

These equations can then be combined in order to obtain the variation related to the conservative variables $\rho, \rho v_i, \rho E$.

The same process is followed to obtain the boundary conditions. However, in this case there is a further complication, because the equations have terms containing the variations ($\delta\rho, \delta v_i, \delta H$). See sub-section 4.3, and eq. 38 for a proper treatment of such terms.

B Formulas for functional variations

B.1 Variation of surface integrals

In the following section the variation of a surface integral (see eq. 42) with respect to surface changes is explicitly derived, in a form useful for computer algebra. Let

$$S = x(u,v)\,\mathbf{i} + y(u,v)\,\mathbf{j} + z(u,v)\,\mathbf{k} \tag{46}$$

be a parametric representation of the integration surface. The surface integral

$$\iint_S f(x,y,z)\,d\sigma$$

can be expressed as a double integral as follows:

$$\int_a^b \int_{\gamma(v)}^{\delta(v)} f(x(u,v),y(u,v),z(u,v))W(u,v)\,du\,dv,$$

where

$$W(u,v) = \sqrt{J_1^2 + J_2^2 + J_3^2}$$

with

$$J_1 = \left|\frac{\partial(y,z)}{\partial(u,v)}\right| \quad J_2 = \left|\frac{\partial(z,x)}{\partial(u,v)}\right| \quad J_3 = \left|\frac{\partial(x,y)}{\partial(u,v)}\right|.$$

Let

$$S' = [x(u,v) + h_1(u,v,\mathbf{a})]\,\mathbf{i} + [y(u,v) + h_2(u,v,\mathbf{a})]\,\mathbf{j} + [z(u,v) + h_3(u,v,\mathbf{a})]\,\mathbf{k}$$

an arbitrary variation of S controlled by a generic parameter vector $\mathbf{a}$. Now we can write:

$$F(\mathbf{a}) = \int_a^b \int_{\gamma(v)}^{\delta(v)} f(x+h_1, y+h_2, z+h_3) W(u,v,\mathbf{a})\, du\, dv.$$

The integration boundaries are unchanged and independent on $\mathbf{a}$. Therefore only the variation of the integrand has to be considered when computing the derivative with respect to a generic element α of $\mathbf{a}$:

$$\begin{aligned}\frac{\partial F(\mathbf{a})}{\partial \alpha} &= \int_a^b \int_{\gamma(v)}^{\delta(v)} \nabla f \cdot \frac{\partial \mathbf{h}(\mathbf{a})}{\partial \alpha} W(u,v,\mathbf{a})\, du\, dv + \\ &\int_a^b \int_{\gamma(v)}^{\delta(v)} f \frac{W(u,v,\mathbf{a})}{\partial \alpha}\, du\, dv.\end{aligned}$$

If f depends explicitly on $\mathbf{a}$, then a further term has to be considered, and the above equation becomes:

$$\begin{aligned}\frac{\partial F(\mathbf{a})}{\partial \alpha} &= \int_a^b \int_{\gamma(v)}^{\delta(v)} \nabla f \cdot \frac{\partial \mathbf{h}(\mathbf{a})}{\partial \alpha} W(u,v,\mathbf{a})\, du\, dv + \\ &\int_a^b \int_{\gamma(v)}^{\delta(v)} f \frac{\partial W(u,v,\mathbf{a})}{\partial \alpha}\, du\, dv + \int_a^b \int_{\gamma(v)}^{\delta(v)} \frac{\partial f}{\partial \alpha} W(u,v,\mathbf{a})\, du\, dv.\end{aligned}$$

Considering that

$$\frac{\partial W(u,v,\mathbf{a})}{\partial \alpha} = \frac{J_1 J_{1_\alpha} + J_2 J_{2_\alpha} + J_3 J_{3_\alpha}}{\sqrt{J_1^2 + J_2^2 + J_3^2}} = \frac{J_1 J_{1_\alpha} + J_2 J_{2_\alpha} + J_3 J_{3_\alpha}}{J_1^2 + J_2^2 + J_3^2}\, W = HW$$

we finally have:

$$\begin{aligned}\frac{\partial F(\mathbf{a})}{\partial \alpha} &= \int_a^b \int_{\gamma(v)}^{\delta(v)} \nabla f \cdot \frac{\partial \mathbf{h}(\mathbf{a})}{\partial \alpha} W(u,v,\mathbf{a})\, du\, dv + \\ &\int_a^b \int_{\gamma(v)}^{\delta(v)} f H(u,v,\mathbf{a}) W(u,v,\mathbf{a})\, du\, dv + \int_a^b \int_{\gamma(v)}^{\delta(v)} \frac{\partial f}{\partial \alpha} W(u,v,\mathbf{a})\, du\, dv.\end{aligned}$$

It can be shown (see next sub-section) that H does not depend on the particular parametric representation chosen, and it is therefore possible to write, finally:

$$\frac{\partial F(\mathbf{a})}{\partial \alpha} = \iint_S \nabla f \cdot \frac{\partial \mathbf{h}}{\partial \alpha}\, d\sigma + \iint_S f H\, d\sigma + \iint_S f_\alpha\, d\sigma.$$

In terms of variation of $\mathbf{h}$ the above formula can be expressed as:

$$\delta F = \iint_S \nabla f \cdot \delta \mathbf{h}\, d\sigma + \iint_S f \delta \boldsymbol{\mathcal{H}}_{\mathbf{h}}\, d\sigma + \iint_S \delta f_{\mathbf{h}}\, d\sigma$$

with

$$\mathcal{H}_j = \frac{J_i \delta J_{i h_j}}{J_1^2 + J_2^2 + J_3^2}.$$

Finally, in terms of variation of a generic function g, we have:

$$\delta F = \iint_S \nabla f \cdot \frac{\partial \mathbf{h}}{\partial g}\, \delta g\, d\sigma + \iint_S f \boldsymbol{\mathcal{H}} \cdot \delta \boldsymbol{\mathcal{H}}_g\, d\sigma + \iint_S \frac{\partial f}{\partial g}\, \delta g\, d\sigma \tag{47}$$

B.2 Invariance of $\mathcal{H}$ with the parametric surface representation

To demonstrate the invariance of $\mathcal{H}_j$ it is sufficient to show that

$$\frac{J_i(s,t)}{J_i(u,v)} = u_s v_t - u_t v_s$$

and

$$\frac{\dfrac{\partial J_i(s,t,\mathbf{h})}{\partial h_j}}{\dfrac{\partial J_i(u,v,\mathbf{h})}{\partial h_j}} = u_s v_t - u_t v_s$$

This can be achieved with simple algebraic passages:

$$\frac{\partial(y,z)}{\partial(u,v)} = \begin{bmatrix} \frac{\partial}{\partial u}\,\mathrm{y}(u,v) & \frac{\partial}{\partial v}\,\mathrm{y}(u,v) \\ \frac{\partial}{\partial u}\,\mathrm{z}(u,v) & \frac{\partial}{\partial v}\,\mathrm{z}(u,v) \end{bmatrix}$$

$$\frac{\partial(u,v)}{\partial(s,t)} = \begin{bmatrix} \frac{\partial}{\partial s}\,\mathrm{u}(s,t) & \frac{\partial}{\partial t}\,\mathrm{u}(s,t) \\ \frac{\partial}{\partial s}\,\mathrm{v}(s,t) & \frac{\partial}{\partial t}\,\mathrm{v}(s,t) \end{bmatrix}$$

$$\frac{\partial(y,z)}{\partial(s,t)} = \frac{\partial(y,z)}{\partial(u,v)} \cdot \frac{\partial(u,v)}{\partial(s,t)} = \begin{bmatrix} \frac{\partial}{\partial u}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{u} + \frac{\partial}{\partial v}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{v} & \frac{\partial}{\partial u}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{u} + \frac{\partial}{\partial v}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{v} \\ \frac{\partial}{\partial u}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{u} + \frac{\partial}{\partial v}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{v} & \frac{\partial}{\partial u}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{u} + \frac{\partial}{\partial v}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{v} \end{bmatrix}$$

$$\frac{J_i(s,t)}{J_i(u,v)} = -\frac{\partial}{\partial s}\,\mathrm{v}\,\frac{\partial}{\partial t}\,\mathrm{u} + \frac{\partial}{\partial s}\,\mathrm{u}\,\frac{\partial}{\partial t}\,\mathrm{v}$$

$$\frac{\partial J_1(u,v)}{\partial h_j} = \frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial v}\,\mathrm{z} + \frac{\partial}{\partial u}\,\mathrm{y}\,\frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{z} - \frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial u}\,\mathrm{z} - \frac{\partial}{\partial v}\,\mathrm{y}\,\frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{z}$$

$$\frac{\partial\left(\frac{\partial y}{\partial h_j}, z\right)}{\partial(u,v)} = \begin{bmatrix} \frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{y} & \frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{y} \\ \frac{\partial}{\partial u}\,\mathrm{z} & \frac{\partial}{\partial v}\,\mathrm{z} \end{bmatrix}$$

$$\frac{\partial\left(y, \frac{\partial z}{\partial h_j}\right)}{\partial(u,v)} = \begin{bmatrix} \frac{\partial}{\partial u}\,\mathrm{y} & \frac{\partial}{\partial v}\,\mathrm{y} \\ \frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{z} & \frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{z} \end{bmatrix}$$

$$\frac{\partial J_i(s,t,\mathbf{h})}{\partial h_j} = \left|\frac{\partial(y_{h_j},z)}{\partial(u,v)} \cdot \frac{\partial(u,v)}{\partial(s,t)}\right| + \left|\frac{\partial(y,z_{h_j})}{\partial(u,v)} \cdot \frac{\partial(u,v)}{\partial(s,t)}\right| =$$
$$\frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{u}\,\frac{\partial}{\partial v}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{v} + \frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{v}\,\frac{\partial}{\partial u}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{u} - \frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{u}\,\frac{\partial}{\partial v}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{v} - \frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{v}\,\frac{\partial}{\partial u}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{u} +$$
$$\frac{\partial}{\partial u}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{u}\,\frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{v} + \frac{\partial}{\partial v}\,\mathrm{y}\,\frac{\partial}{\partial s}\,\mathrm{v}\,\frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{z}\,\frac{\partial}{\partial t}\,\mathrm{u} - \frac{\partial}{\partial u}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{u}\,\frac{\partial^2}{\partial v\,\partial h_j}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{v} - \frac{\partial}{\partial v}\,\mathrm{y}\,\frac{\partial}{\partial t}\,\mathrm{v}\,\frac{\partial^2}{\partial u\,\partial h_j}\,\mathrm{z}\,\frac{\partial}{\partial s}\,\mathrm{u}$$

$$\frac{\dfrac{\partial J_i(s,t,\mathbf{h})}{\partial h_j}}{\dfrac{\partial J_i(u,v,\mathbf{h})}{\partial h_j}} = -\frac{\partial}{\partial s}\,\mathrm{v}\,\frac{\partial}{\partial t}\,\mathrm{u} + \frac{\partial}{\partial s}\,\mathrm{u}\,\frac{\partial}{\partial t}\,\mathrm{v}$$

B.3 Volume integral variation

The volume integral variation is obtained making explicit reference to the reduction formulas of a volume integral into a triple integral, and applying in a straightforward way the formula for the derivative of a definite integral:

$$\int_{u(t)}^{v(t)} f(x,t)dx = \int_{u(t)}^{v(t)} \frac{\partial}{\partial t} f(x,t)dx + f(v(t),t)\frac{d}{dt}v(t) - f(u(t),t)\frac{d}{dt}u(t)$$

which, applied to a triple integral, gives:

$$\begin{aligned}
\frac{\partial}{\partial t}\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)}\int_{\phi(x,y,t)}^{\psi(x,y,t)} f(x,y,z,t)dz\,dy\,dx = \\
&\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)}\int_{\phi(x,y,t)}^{\psi(x,y,t)} \frac{\partial}{\partial t} f(x,y,z,t)dz\,dy\,dx \\
+ &\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} \left(\frac{\partial}{\partial t}\psi(x,y,t)\right) f(x,y,\psi(x,y,t),t)dy\,dx \\
- &\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} \left(\frac{\partial}{\partial t}\phi(x,y,t)\right) f(x,y,\phi(x,y,t),t)dy\,dx \\
+ &\int_{a(t)}^{b(t)} \left(\frac{\partial}{\partial t}\delta(x,t)\right)\int_{\phi(x,\delta(x,t),t)}^{\psi(x,\delta(x,t),t)} f(x,\delta(x,t),z,t)dzdx \\
- &\int_{a(t)}^{b(t)} \left(\frac{\partial}{\partial t}\gamma(x,t)\right)\int_{\phi(x,\gamma(x,t),t)}^{\psi(x,\gamma(x,t),t)} f(x,\gamma(x,t),z,t)dzdx \\
+ &\left(\frac{d}{dt}b(t)\right)\int_{\gamma(b(t),t)}^{\delta(b(t),t)}\int_{\phi(b(t),y,t)}^{\psi(b(t),y,t)} f(b(t),y,z,t)dz\,dy \\
- &\left(\frac{d}{dt}a(t)\right)\int_{\gamma(a(t),t)}^{\delta(a(t),t)}\int_{\phi(a(t),y,t)}^{\psi(a(t),y,t)} f(a(t),y,z,t)dz\,dy
\end{aligned}$$

Introducing the deformation field $\mathbf{h}(x,y,z)$, the boundary function ψ and ϕ can be written as:

$$\psi(x,y,t) = \psi_0(x,y) + h_3(x,y,\psi_0(x,y),t)$$

$$\phi(x,y,t) = \phi_0(x,y) + h_3(x,y,\phi_0(x,y),t)$$

This allows the identity below:

$$\begin{aligned}
&\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} \left(\frac{\partial}{\partial t}\psi(x,y,t)\right) f(x,y,\psi(x,y,t),t)dy\,dx + \\
&-\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} \left(\frac{\partial}{\partial t}\phi(x,y,t)\right) f(x,y,\phi(x,y,t),t)dy\,dx \quad = \\
&\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} D_4(h_3)(x,y,\psi_0(x,y),t) f(x,y,\psi(x,y,t),t)dy\,dx + \\
&-\int_{a(t)}^{b(t)}\int_{\gamma(x,t)}^{\delta(x,t)} D_4(h_3)(x,y,\phi_0(x,y),t) f(x,y,\phi(x,y,t),t)dy\,dx \quad = \quad \iint_{+FT} \frac{\partial h_3}{\partial t}\, f dy\,dx
\end{aligned}$$

Reasoning in a similar way for the remaining couples of integrals, we can finally write:

$$\frac{\partial}{\partial t}\iiint_T f\,dV = \iiint_T \frac{\partial f}{\partial t}\,dV + \iint_{+FT} \frac{\partial h_1}{\partial t}\, f dy\,dz + \iint_{+FT} \frac{\partial h_2}{\partial t}\, f dz\,dx + \iint_{+FT} \frac{\partial h_3}{\partial t}\, f dx\,dy$$

That, in terms of surface integrals, becomes:

$$\frac{\partial}{\partial t}\iiint_T f\,dV = \iiint_T \frac{\partial f}{\partial t}\,dV + \iint_{FT} f\,\frac{\partial \mathbf{h}}{\partial t}\cdot\mathbf{n}\,d\sigma$$

Finally, in terms of variation of a generic function g, we have:

$$\delta F = \iiint_T \frac{\partial f}{\partial g}\delta g\,dV + \iint_{FT} f\,\frac{\partial \mathbf{h}}{\partial g}\cdot\mathbf{n}\delta g\,d\sigma \tag{48}$$

B.4 Application of divergence theorem to terms involving crossed derivatives

The divergence theorem

$$\iiint_\Omega \nabla\cdot\mathbf{b}\,d\Omega = \iint_\Sigma \mathbf{b}\cdot\mathbf{n}\,d\Sigma$$

can be applied in two different ways to transform the following volume integral

$$\iiint_\Omega v_{xy}\,d\Omega$$

with v a generic scalar function continuous in Ω with its first- and second-order partial derivatives, such that $v_{xy} = v_{yx}$. It is, indeed, possible to write v_{xy} either as $\nabla\cdot(0, v_x, 0)$ or as $\nabla\cdot(v_y, 0, 0)$. This leads immediately to the following identity:

$$\iiint_\Omega v_{xy}\,d\Omega = \iint_\Sigma v_x n_2\,d\Sigma = \iint_\Sigma v_y n_1\,d\Sigma$$

with $\mathbf{n} = (n_1, n_2, n_3)$.

This identity can be used to transform some adjoint boundary conditions. This operation, indeed, can be useful to restore symmetry in the adjoint boundary conditions automatically derived.

References

[1] M. Drela. Xfoil: An analysis and design system for low reynolds number airfoils. In *Conference on Low Reynolds Number Airfoil Aerodynamics*. University of Notre Dame, June 1989.

[2] M. Drela and M.B. Giles. Viscous-inviscid analysis of transonic and low reynolds number airfoils. *AIAA Journal*, 25(10):1347–1355, October 1987.

[3] A. Jameson. Aerodynamic design via control theory. Technical Report ICASE Report 88-64, NASA Langley Research Center, 1988.

[4] A. Jameson, L. Martinelli, and N. A. Pierce. Optimum aerodynamic design using the Navier-Stokes equations. *TCFD*, (10):213–237, 1998.

[5] A. Iollo and L. Zannetti. Trapped vortex optimal control by suction and blowing at the wall. *EJM/B Fluids*, 20:417–430, 2001.

[6] A. Iollo, M. Ferlauto, and L. Zannetti. An aerodynamic optimization method based on the inverse problem adjoint equations. *Journal of Computational Physics*, 173(1):87–115, 2001.

[7] N.A. Pierce and M.B. Giles. Adjoint recovery of superconvergent functionals from approximate solutions of partial differential equations. Technical Report Report no. 98/18, Oxford Computing Laboratory, 1998.

[8] D. Redfern. *Maple V Handbook - Release 4*, 1996. ISBN 0-387-94538-5.

[9] P. Musgrave, D. Pollney, and K. Lake. GRTensorII release 1.50 for MapleV releases 3 and 4 — introduction and overview. Technical report, Queen's University, Kingston, Ontario, July 1996.

[10] E. S. Cheb-Terrab. Maple procedures for partial and functional derivatives. *Computer Physics Communications*, 79:409–424, 1994.

[11] E. Arian and M. D. Salas. Adjoint formulation using auxiliary boundary equations. In Charles-Henri Bruneau, editor, *Lecture Notes in Physics*, volume 515, pages 355–360, Berlin Heidelberg, 1998. Springer-Verlag.

[12] A. P. Giotis, K. C. Giannakoglou, and Jacques Périaux. A reduced-cost multi-objective optimization method based on the pareto front technique, neural networks and pvm. In *Proceedings of ECCOMAS 2000 Conference*, Barcelona, Spain, September 11–14 2000.

[13] T. Poggio and F. Girosi. A theory of networks for approximation and learning. Technical Report AIM–1140, Artificial Intelligence Laboratory – MIT, 1989.

[14] T. Poggio and F. Girosi. Extension of a theory of networks for approximation and learning: Dimensionality reduction and clustering. In *Proceedings Image Understanding Workshop*, pages 597–603, Pittsburgh, Pennsylvania, September 1990.

[15] T. Poggio and F. Girosi. Hyperbf: A powerful approximation technique for learning. In P. H. Winston and S. A. Shellard, editors, *Artificial Intelligence at MIT: Expanding Frontiers*, pages 270–285, Cambridge, Massachusetts, 1990. The MIT Press.

[16] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, (247):978–982, 1990.

[17] Kalyanmoy Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[18] Alessandro Vicini and Domenico Quagliarella. Inverse and direct airfoil design using a multiobjective genetic algorithm. *AIAA Journal*, 35(9):1499–1505, September 1997.

[19] Domenico Quagliarella and Alessandro Vicini. GAs for aerodynamic shape design II: multiobjective optimization and multi-criteria design. In *Lecture Series 2000-07, Genetic Algorithms for Optimisation in Aeronautics and Turbomachinery*, Belgium, May 2000. Von Karman Institute.

[20] Hans-Paul Schwefel. *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology. John Wiley & Sons, New York, U.S.A., 1994.

[21] Thomas Bäck and Hans-Paul Schwefel. Evolutionary algorithms: Some very old strategies for optimization and adaptation. In D. Perret-Gallix, editor, *New Computing Techniques in Physics Research II*, Singapore, 1992. World Scientific.

This page has been deliberately left blank

Page intentionnellement blanche